Visible Surface Reconstruction With Accurate Localization of Object Boundaries

Federico Pedersini, Augusto Sarti, and Stefano Tubaro

***Abstract*—A common limitation of many techniques for 3-D reconstruction from multiple perspective views is the poor quality of the results near the object boundaries. The interpolation process applied to "unstructured" 3-D data ("clouds" of non-connected 3-D points) plays a crucial role in the global quality of the 3-D reconstruction. In this paper, we present a method for interpolating unstructured 3-D data, which is able to perform a segmentation of such data into different data sets that correspond to different objects. The algorithm is also able to perform an accurate localization of the boundaries of the objects. The method is based on an iterative optimization algorithm. As a first step, a set of surfaces and boundary curves are generated for the various objects. Then, the edges of the original images are used for refining such boundaries as best as possible. Experimental results with real data are presented for proving the effectiveness of the proposed algorithm.**

I. Introduction

AMONG the available methods for image-based 3-D scene reconstruction, a leading role is played by stereometric techniques based on feature correspondences. Such methods recover the 3-D coordinates of object features by detecting, matching and back-projecting homologous image features on two or more perspective views, taken from different viewpoints. One drawback of such methods, however, is that they are only able to reconstruct those portions of the surface that are visible from at least two viewpoints. If the acquisition system is placed in front of the scene, it is reasonable to model the surface through a "depth map." This $2(1/2)$-D representation will exhibit discontinuities in the proximity of surface occlusions, which normally correspond to the boundaries between different objects.

If, on one hand, the depth map is expected to be discontinuous at the object boundaries, on the other hand, correspondence-based reconstruction techniques often fail to provide accurate information in the vicinity of such boundaries. In such regions, in fact, the 3-D data turns out to be very sparse and often affected by significant errors and artifacts. The poor characterization of 3-D data near the object boundaries causes surface interpolators to perform poorly in areas where the accuracy is of utmost importance. In fact, even if the boundary regions are only a small portion of the whole scene, their importance is crucial, as they carry the most significant information on the object shape.

Manuscript received March 15, 1999; revised September30, 1999. This paper was recommended by Guest Editor M. G. Strintzis.

The authors are with the Image and Sound Processing Group, Dipartimento di Elettronica e Informazione, Politecnico di Milano, 20133 Milano, Italy (e-mail: pedersin@elet.polimi.it; sarti@elet.polimi.it; tubaro@elet.polimi.it).

Publisher Item Identifier S 1051-8215(00)02017-6.

Several methods have been developed for interpolating sparse 3-D data. Among such methods, it is important to mention [1], which implements a modification of the thin-plate spline algorithm. Through this methods it is possible to model "cuts" and "creases" of the plate. Surface cuts model depth discontinuities at the object's boundaries, while creases model discontinuities of the first derivative of the surfaces (edges and sharp rims). Terzopoulos [2], [3] proposed a method for jointly determining the best interpolating surface and the location of a set of curves where cuts and folds take place. This method, however, is based on the minimization of a functional which requires a rather heavy computational load. Furthermore, when the 3-D data is extracted from strongly converging perspective views, the quality of the 3-D information near the objects boundaries turns out to be quite poor. In this situation, this method does not have enough information to reconstruct the objects silhouette with an adequate accuracy.

In this paper, we present a method for interpolating unstructured 3-D data, which is able to segment the data into sets that correspond to different objects and, at the same time, to perform an accurate localization of the object boundaries. Our approach begins with an iterative optimization process, which maximizes a functional that is similar to the one defined by Terzopoulos [2], and returns a set of surfaces that model the objects and their boundaries. A segmentation algorithm is then applied to the perspective projection of the resulting surface. This algorithm partitions this surface into sub-surfaces of continuous depth, which are likely to correspond to different objects and, for each one of them, it determines a closed curve that encircles it and approximates the *boundary* of the object. The last step of the procedure uses the luminance/color edges for refining the position of such boundaries. This is accomplished through a recursive boundary update algorithm, whose aim is to "pull' the closed curves toward the image projection of the objects silhouettes (which are visible as luminance or color edges along the object boundaries).

The paper is organized as follows. The next section describes the adopted data representation and the general formalization of the visible surface reconstruction problem. Section III describes the two functional blocks that constitute the core of the proposed technique: the combined interpolation/segmentation algorithm and the accurate boundary localization. In Section IV, the results of some experiments conducted on real sequences (acquired with trinocular systems based on standard TV-resolution cameras) are presented.

II. Preliminaries

A. Projective Representation of the Depth Map

A stereometric imaging system that acquires the front views of a scene can determine the 3-D coordinates of a scene feature only if it is visible from at least two viewpoints. As a consequence, the surface regions that can be reconstructed are generally a subset of the visible regions. for this reason, the reconstructed surface can be represented as an explicit function of the form $z = u(x, y)$, where x and y are the coordinates of a *reference plane* that is usually chosen as parallel to one of the image planes, and z represents the distance of the surface from the (x, y) plane or from a given viewpoint.

A typical way to represent depth maps is the *orthographic representation*, shown in Fig. 1(a). The surface reference frame $O(x, y, z)$ is Cartesian and lies ideally behind the scene, while z represents the "height" of the surface at the point (x, y) of the reference plane. Such a representation turns out to be quite intuitive, as the observed scene is represented like a 3-D high-relief, whose base coincides with the reference plane. We should keep in mind, however, that the actual imaging process consists, of a perspective projection of the scene. Consequently, the 3-D shape of the visible scene surfaces would be better described by a "projective" depth map originating from the reference viewpoint. Such a depth map can be thought of as a bundle of depth rays that meet at the camera's optical center, each one intersecting first the image plane and then the scene at a distance D (depth) from the optical center, as shown in Fig. 1(b). This *projective representation* of depth maps has the advantage of guaranteeing a certain consistency between visible and reconstructible surfaces, as the mapping between image points of the reference view and depth points of the surface is one-to-one. As we can see in Fig. 1(a), this consistency is generally not guaranteed by an orthographic depth map, as there could be portions of the reconstruction's domain that correspond either to regions that are not visible from the reference viewpoint or to more than one region of the visible surface.

Furthermore, a projective representation of the depth map results to be particularly suitable in the case of trinocular vision. In fact, when one of the three cameras of the acquisition system is placed approximately between the other two, each point of the surfaces that can be reconstructed through stereo correspondences using any pair of cameras will be visible from this camera. For instance, if the three viewpoints were collinear, the union of the surfaces regions that could be reconstructed from any pair of cameras would be contained in the set of surfaces visible from the middle viewpoint.[1] For this reason, it is particularly convenient to adopt a projective depth-map originating from the optical center of the middle camera, which becomes the *reference viewpoint*.

[1]One case that could raise doubts on this statement is the one of two objects, one of which occludes the other. In this case, the two side views could see the occluded object while the middle one could not. However, ordering constraints in binocular-matching processes would always prevent the second surface from being reconstructed using feature matching.

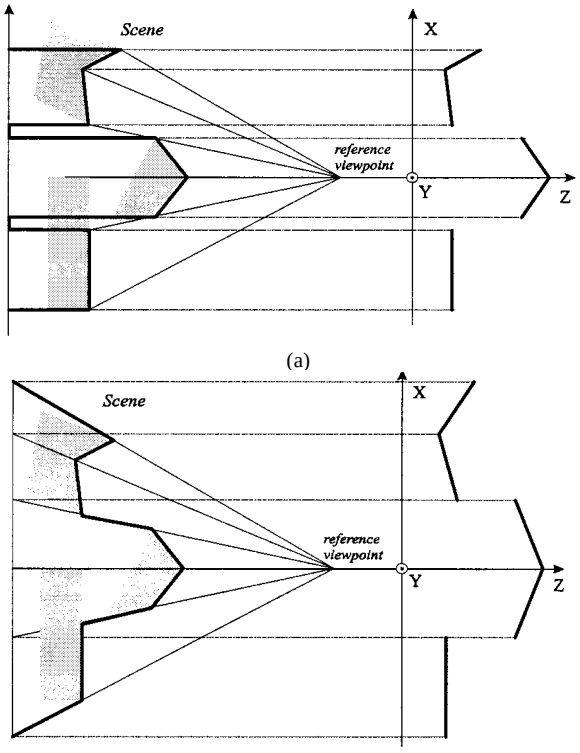


Fig. 1. Traditional and proposed representation of the depth map. (a) *Orthographic representation*: the depth u is represented in a 3-D Cartesian reference $O(x, y, z)$ where $z = u(x, y)$ and (x, y) is parallel to the image plane. (b) *Projective representation*: the depth u is perspectively mapped onto the image plane. The depth value defined for each image point is the distance of the viewed surface from the projection center (the viewpoint) of the reference view.

B. Problem Formalization

Let us consider the portion $\mathcal{S}_v$ of the scene surface $\mathcal{S}$ that is visible from the reference viewpoint. As explained in the previous section, $\mathcal{S}_v$ can be modeled with a projective depth map of the form $z = D(x, y)$, where z is the distance (from the reference viewpoint) of a scene point P that projects onto the point (x, y) of a plane π called *reference plane*. The extension of the domain of (x, y) is chosen in such a way to cover the whole visual field of the acquisition system. The visible surfaces $\mathcal{S}_v$ exhibit discontinuities wherever the depth-ray of the projective depth map abruptly goes from one object to another one in a situation of partial occlusion. In this case, the occluding surface will be conventionally referred to as *foreground*, while the occluded one will be considered *background*. The available input data is a set of 3-D measurements on the visible surfaces, which can be thought of as a sparse, irregular and noisy sampling of $\mathcal{S}_v$. In conclusion, a 3-D reconstruction of good quality should be made of a set of surfaces, each of which interpolates as best as possible the sparse 3-D data that pertain a different object of the scene. The projective depth map will thus be partitioned in such a way to exhibit discontinuities at the object's boundaries. Each one of the surfaces, however, will be assumed smooth, unless the object that it represents exhibits a sharp edge, in which case the surface will be allowed a crease (tangent plane's discontinuity).

With the above requirements in mind, it is reasonable to think of a discontinuity-preserving thin-plate spline as the best approach to solve the visible surface reconstruction problem. A thin plate, in fact, is characterized by its own cohesion force and, when pulled toward the 3-D sparse data, tends to model the visible surfaces in a physical fashion. However, in order to model cuts (at object boundaries and occlusions) and creases (along sharp edges), we need to be able to define *cutting* and *folding* contours on the thin plate. This approach was adopted by Terzopoulos [2], Mallet [1] and Blake and Zisserman [3], who formalized the problem as a variational one. In [3], the concept of *controlled-continuity interpolator* was introduced, which allowed the interpolating surface to be more elastic (smoother) or more rigid (more prone to tearing), depending on the nature of the data. The approach that we propose in this article, which is presented in what follows, represents a generalization of [2].

Mathematical model—Let us consider a *continuous thin plate* represented by an explicit surface $z = u(x, y)$, which is "pulled" by a set of 3-D points $P_i = (x_i, y_i, z_i)$ of known coordinates. If we want this plate to minimize the mean square distance from the 3-D data while minimizing its internal energy, then we can define a functional of the form

$$\varepsilon'(u) = \iint_{\Re^2} \left(u_{xx}^2 + 2u_{xy} + u_{yy}^2\right) dx\, dy + \alpha \sum_{P_i} (u(x_i, y_i) - d_i)^2 \quad (1)$$

where u_{xx}, u_{xy} and u_{yy} are the second-order derivatives of $u(x, y)$, d_i is the distance of P_i from the surface (along the z axis). Through the first term of (1), which represents the internal energy of the interpolating surface, we control the local curvature, therefore its minimization tends to smoothen the surface [6]. The second term controls "how well" the 3-D data are honored. Such two normally contrasting needs are balanced through the weight coefficient α. This *continuous thin-plate model* represents a good solution to the 3-D data interpolation problem when the original surface does not exhibit either creases or depth discontinuities.

In order to allow the surface to model creases, we should ignore the internal energy term and allow discontinuities of the surface gradient. A physical model suitable for the interpolation surfaces with creases is the *elastic membrane*. In this case, the regularization term of the corresponding functional ε' is expressed as a function of the first-order derivatives of u (magnitude of the gradient)

$$\varepsilon''(u) = \iint_{\Re^2} \left(u_x^2 + u_y^2\right) dx\, dy + \alpha \sum_{P_i} (u(x_i, y_i) - d_i)^2.$$

Similarly, as in the case of the thin-plate model, in order for an elastic membrane to be able to model also depth discontinuities, the regularization term of the functional should be neglected along such lines.

Taking into account for these considerations, Terzopoulos [2] has come to the definition of a functional which accounts for both discontinuities and creases through the introduction of two regularization terms

$$\varepsilon(u, \omega) = \iint_{\Re^2} \rho(x, y) \left\{\tau(x, y) \left(u_{xx}^2 + 2u_{xy} + u_{yy}^2\right) + [1 - \tau(x, y)] \left(u_x^2 + u_y^2\right)\right\} dx\, dy + \alpha \sum_{P_i} (u(x_i, y_i) - d_i)^2 + D(\omega) \quad (2)$$

where $u_x, u_y, u_{xx}, u_{yy}, u_{xy}$ are the first- and second-order partial derivatives of $u(x, y)$, $\rho(x, y)$ is the discontinuity map, which is equal to one everywhere, except in correspondence of surface discontinuities, where it is set to zero and, similarly, $\tau(x, y)$ represents the map of the surface creases, as it is set to zero in correspondence of them. Notice that, in order for this formulation to make sense, we need to assume that the curves of discontinuity have non-zero width, otherwise the set in which $[1 - \tau(x, y)]$ is non-zero in (2) would have zero measure. This problem, however, has a trivial solution in the discrete domain, which is our case. The functional also presents an additional term that depends on ω, which is the set of discontinuities and crease curves. The term $D(\omega)$ actually measures the total extension of the discontinuity curves. The introduction of this term is necessary, as it prevents the functional from producing a degenerate solution, such as one in which all the points are classified as cuts and creases ($\rho(x, y) = \tau(x, y) = 0$, which are points where the surface is not defined), except for those of the data set, where we have $u(x, y) = d(x, y)$. In other words, the term $D(\omega)$ prevents the interpolated surface from "collapsing" into the point cloud. There are several ways to define $D(\omega)$; a very simple but efficient one is to choose it so that it will measure the total length of the discontinuities and crease lines.

Notice that the minimization of the functional (2) is made with respect to both u and ω, which means that the algorithm returns the best surface $u(x, y)$ that interpolates the assigned 3-D points, as well as the maps of discontinuity and surface creases $\rho(x, y)$ and $\tau(x, y)$, respectively, which are assumed visible when the scene is observed from the *reference* viewpoint.

C. Computational Issues

The functional described in Section II-B addresses the variational reconstruction problem in an accurate and elegant fashion. Operatively speaking, however, a direct minimization of this functional is not such a straightforward task. For example, the choice of the discrete operators to be used for computing the first- and second-order derivatives of $u(x, y)$ gives rise to some difficulties in the discretization process. Other problems are encountered with the definition of the maps $\rho(x, y)$ and $\tau(x, y)$, and of $D(\omega)$ in the discrete domain. Another non-trivial complication arises from the nature of the considered functional. In fact, (2) is a non-convex functional with multiple relative minima, therefore its minimization must necessarily be performed iteratively. Among the many optimization algorithms, *simulated annealing* seems to be the only one that could guarantee a successful global optimization. Considering the complexity of our problem, however, this approach is unfeasible due to the excessive computational cost. It is thus necessary to develop an optimization technique that

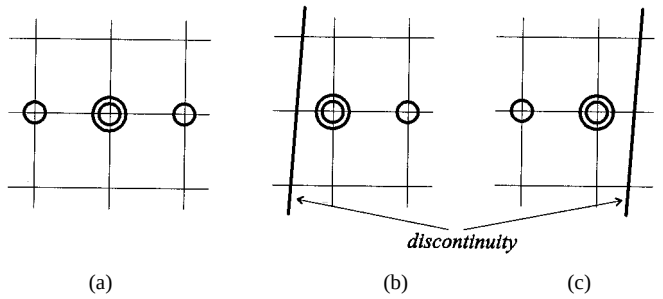


Fig. 2. Impulse-response masks for computing the first-order surface derivatives. (a) Case of a continuous surface. (b) Neighboring discontinuity on the left. (c) Neighboring discontinuity on the right.

reaches a correct solution with a reasonable computational effort. In this section, we present our approach to both the discretization and the minimization problems.

1) Discretization of the Functional: In order to discretize the functional (2), we need to approximate integrals with sums and derivatives with finite differences. As far as the integral is concerned, its direct discretization would be quite straightforward if there were not the discontinuity curves to take into account. The simplest way to do so is to discretize the location of the discontinuities through the definition of two discrete (pixel-resolution) maps $\rho(i,j)$ and $\tau(i,j)$. The choice of one-pixel-wide discontinuity curves greatly simplifies the measurement of the extension of the discontinuities $D(\omega)$. In fact, as this measurement corresponds to the total length of such curves, it can be carried out in a rather accurate way by simply counting the number of pixels where the map is zero.

The discretization of the derivatives, in case of smooth surfaces, is quite a straightforward task. In the proximity of the discontinuities, however, the situation becomes more complex. On one hand, the discontinuity maps $\rho(i,j)$ and $\tau(i,j)$ are defined in order to cancel the contribution of internal energy terms (which are the only ones that depend on the surface derivatives) at the discontinuity curves. On the other hand, we should remember that the computation of the derivatives is implemented through digital filters, whose impulse response tends to invade the discontinuity when in its proximity. This often results in an incorrect response peak. Mallet [1] and Terzopoulos [2] proposed a modification of the difference operators near the discontinuities, in such a way to evaluate curvatures and slopes without crossing the discontinuity line. Examples of impulse response masks for such operators are shown in Fig. 2. For example, in the neighborhood of a depth discontinuity on the left, the mask (b) is applied instead of (a), when the discontinuity is on the right, the mask (c) replaces (a).

The behavior of the interpolated surface near and on the discontinuities strongly depends on the choice of the kernels. In the example of Fig. 2, for example, no depth value is defined in the exact correspondence of the lines of discontinuity. This behavior can be controlled through a proper definition of the kernels used near the discontinuities. The use of different and properly shaped kernels in the different situations constitutes the algorithmic device that allows modification of the "region of influence" of the interpolation, depending on the presence and on the type of discontinuities. This is the key feature of any discontinuity-preserving interpolation scheme.

Taking the above considerations into account, the discretized version of the functional $\mathcal{E}(u,w)$ can be defined as

$$\begin{aligned}\mathcal{E}(u,w) = &\sum_i \sum_j \rho(i,j)\tau(i,j)\left[u_{xx}^2(i,j) + 2u_{xy}^2(i,j) + u_{yy}^2(i,j)\right] \\ &+ \sum_i \sum_j \rho(i,j)[1-\tau(i,j)]\left[u_x^2(i,j) + u_y^2(i,j)\right] \\ &+ \sum_k \alpha(u(i_k,j_k) - d_k)^2 + D(\omega) \qquad (3)\end{aligned}$$

where the definition of the differential operators depends on the presence of a discontinuity in its neighborhood, on its position and on its type. The first-order derivative operator $u_x(i,j)$, for example, is defined as shown in the equation at the bottom of the page.

2) Localization of the Discontinuity Lines: The goal of the optimization process is to determine a surface u and a set of discontinuity contours ω that minimize the discrete functional (3). Notice that the localization of the discontinuities must use the 3-D point coordinates as the only source of information. Consequently, although the variational problem could be solved just through a direct search of the global minimum of the functional, reasons of computational efficiency suggest to first roughly estimate the discontinuity contours, and then let the global optimization algorithm take care of refining this information.

A rough estimate of the discontinuity contours can be obtained by determining those surface regions where the traction force "exerted" by the 3-D data points is so strong as to "break" the plate. The magnitude of the surface gradient $|\nabla u(x,y)|$ can be considered as a reliable measure of this force, as it describes the local surface slope. Criteria for the localization of probable discontinuities, based on the localization of significant peaks in the magnitude of the gradient of the depth function $u(x,y)$, are well known in literature. In fact, if we interpret $u(x,y)$ as a luminance profile, the problem becomes that of the *edge detection*, which has been abundantly studied in the literature.

As an immediate consequence of the above consideration, we chose to perform a first rough localization of the discontinuities using Canny's edge-detection algorithm [17]. As this edge detector is designed for 2-D surfaces that take value on regular grids, we use as input an interpolated version of the sparse 3-D data. However, some variations on the classical Canny's edge-

$$u_x(i,j) = \begin{cases} u(i,j+1) - u(i,j), & \text{depth discontinuity on the left:} \rho(i,j-1) = 0 \\ u(i,j) - u(i,j-1), & \text{depth discontinuity on the right:} \rho(i,j+1) = 0 \\ \frac{u(i,j+1) - i(i,j-1)}{2}, & \text{otherwise} \end{cases}.$$

detection approach are necessary, due to the type of pre-processing and the different nature of the input data. In particular, in order to achieve a significant quality in the first discontinuity detection, we introduced the following modifications:

1) As 2-D input profile is the result of an interpolation, its spatial frequency content is very low pass and the signal-to-noise ratio (SNR) is significantly higher than in the case of luminance profiles. Consequently, the Gaussian 2-D pre-filtering turned out to be unnecessary or even disturbing, as it reduces the quality of the discontinuity detection. The gradient operator is thus directly applied to the unfiltered depth map.
2) If the detected edge contours exhibit an excessive curvature in some points where the edge is expected to be smooth, such irregularities are back-propagated onto the depth map. The thin-plate interpolation will thus exhibit annoying artifacts, such as jagged object boundaries (see, for example, Fig. 3). In order to avoid such problems, the last step of the edge-detection algorithm consists of a sophisticated recursive edge-following process [18], [21]. This operation is based on both the orientation of the gradient maxima and on the already detected edges. The global edge detection algorithm guarantees edges of controlled smoothness, which it allows us to avoid the above artifacts.

III. The Algorithm

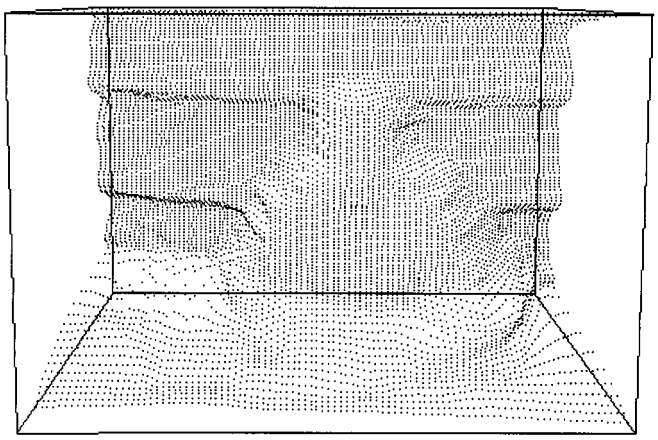
As anticipated in the introduction, aim of the proposed method is the interpolation of unstructured 3-D data, the segmentation of the interpolated surfaces into parts, each representing a different object, and finally the accurate localization of the boundaries of such objects.

The interpolation and segmentation is performed at the same time, through the maximization of the discrete functional (3), leading to a set of subsurfaces, where each sub-surface interpolates the corresponding cloud of 3-D points. The algorithm partitions this surface into sub-surfaces of continuous depth, which are likely to correspond to different objects and, for each one of them, it determines a closed curve that encircles it and approximates the *boundary* of the object. Because of the usual poorness of 3-D information in the vicinity of the boundaries of the objects, these sub-surfaces generally present a quite rough and irregular boundary, sometimes also significantly distant from the actual occlusion boundary.[2]

In order to better localize such boundaries, the only information that could be exploited is their actual visibility in the original images of the scene. If a boundary is visually recognizable in the images, it means that there is enough information, in the luminance/chrominance profiles, to localize its projection. For this reason, the second step of the procedure uses the luminance/color edges for refining the position of such boundaries. This is accomplished through a recursive boundary update algorithm, whose aim is to "pull" the closed curves toward the image projection of the objects silhouettes (which are visible as luminance or color edges along the object boundaries).

[2]Since the position of occlusion boundaries is viewpoint-dependent, the boundary seen from the *reference viewpoint* is considered, if not otherwise specified.

(a)



(b)

Fig. 3. (a) Original view. (b) Corresponding point-set. Where the detected edges are jagged, the irregularities are propagated onto the depth map, originating annoying artifacts at object borders.

In the following, these two steps are described in details.

A. Step 1: The Segmentation/Interpolation Algorithm

As a result of the above considerations, the segmentation/interpolation algorithm is organized as in the block diagram of Fig. 4, which contains the following functional blocks.

1) Interpolation based on the minimization of the functional $\mathcal{E}(u,\omega)$ [see (3)], with respect to u. The first interpolation step is performed with a single thin plate with no discontinuities.
2) Rough estimation of the discontinuity contours $\omega = \{\rho,\tau\}$: estimation of $\rho(i,j)$ through Canny's edge detection.
3) Adjustment of the detected discontinuity curves to the boundaries of the corresponding foreground objects.
4) Segmentation into subsurfaces corresponding to different objects, followed by a separate interpolation of each one of them, using the boundaries determined at Step 3.

We will now discuss each one of the above steps in detail.

Surface Initialization—In order for the interpolator to work, the depth map must be defined over the whole domain (refer-

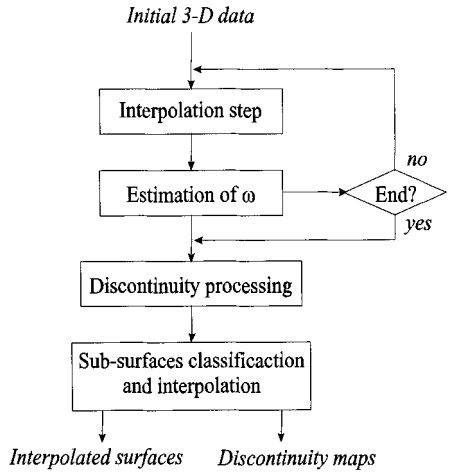


Fig. 4. Flow chart of the proposed algorithm.

ence plane). In order to do so, we fit a single rigid and continuous thin plate over the point-set. This operation is easily performed through the minimization of the functional (3), assuming $\rho(i,j) = \tau(i,j) = 1, \forall i,j$ (smooth surface).

One other important reason for performing pre-interpolation is that it allows us to easily detect and eliminate any outliers in the input point-set. Such points, in fact, are normally present in point-sets that are obtained with automatic 3-D reconstruction procedures. In fact, matching errors typically result in 3-D points of unpredictable coordinates, which can be easily detected while fitting the thin plate.

Interpolation—As already stated above, the interpolation step consists of the minimization of the functional (3), with respect to $u(i,j)$. Apart from the first iteration, the discontinuity maps $\rho(i,j)$ and $\tau(i,j)$ are given as an input. The minimization is carried out through relaxation, by iteratively updating each point as follows:

$$u_{i,j}^{(n)} = u_{i,j}^{(n-1)} - \frac{\lambda}{T}\frac{\partial \mathcal{E}(u,\omega)}{\partial u_{i,j}} \tag{4}$$

where the weight λ is normalized by the number T of samples used for computing $\partial \mathcal{E}(u,\omega)/\partial u_{i,j}$.

As described in last section, the gradient term $\partial \mathcal{E}(u,\omega)/\partial u_{i,j}$ and, of course, T, differ from point to point depending on the configuration of the discontinuities in the point's neighborhood. When no discontinuities are in the neighborhood of the point $u_{i,j}$, the gradient of the functional $\mathcal{E}(u,\omega)$ with respect to $u(i,j)$ assumes the form

$$\begin{aligned}
\frac{\partial \mathcal{E}(u,\omega)}{\partial u_{i,j}} &= 2\rho_{i,j}(1-\tau_{i,j})(u_{i,j} - u_{i,j+1}) \\
&\quad - 2\rho_{i,j-1}(1-\tau_{i,j-1})(u_{i,j-1} - u_{i,j}) \\
&\quad + 2\rho_{i,j}(1-\tau_{i,j})(u_{i,j} - u_{i-1,j}) \\
&\quad - 2\rho_{i+1,j}(1-\tau_{i+1,j})(u_{i+1,j} - u_{i,j}) \\
&\quad - 4\rho_{i,j}\tau_{i,j}(u_{i,j-1} + u_{i,j+1} - 2u_{i,j}) \\
&\quad + 2\rho_{i,j+1}\tau_{i,j+1}(u_{i,j} + u_{i,j+2} - 2u_{i,j+1}) \\
&\quad + 2\rho_{i,j-1}\tau_{i,j-1}(u_{i,j-2} + u_{i,j} - 2u_{i,j-1}) \\
&\quad - 4\rho_{i,j}\tau_{i,j}(u_{i-1,j} + u_{i+1,j} - 2u_{i,j}) \\
&\quad + 2\rho_{i+1,j}\tau_{i+1,j}(u_{i,j} + u_{i+2,j} - 2u_{i+1,j}) \\
&\quad + 2\rho_{i-1,j}\tau_{i-1,j}(u_{i-2,j} + u_{i,j} - 2u_{i-1,j}) \\
&\quad - 4\rho_{i,j}\tau_{i,j}(-u_{i,j} - u_{i+1,j+1} + u_{i,j+1} + u_{i-1,j}) \\
&\quad - 4\rho_{i-1,j-1}\tau_{i-1,j-1}(-u_{i-1,j-1} - u_{i,j} \\
&\quad + u_{i-1,j} + u_{i-2,j-1}) + 4\rho_{i,j-1}\tau_{i,j-1} \\
&\quad \times (-u_{i,j-1} - u_{i+1,j} + u_{i,j} + u_{i-1,j-1}) \\
&\quad + 4\rho_{i+1,j}\tau_{i+1,j}(-u_{i+1,j} - u_{i+2,j+1} \\
&\quad + u_{i+1,j+1} + u_{i,j}) + 2\alpha(u_{i,j} - d_{i,j}).
\end{aligned}$$

The term λ decides on the convergence speed and on the accuracy of the final solution. A small value of λ results in a slow convergence while a higher value causes the vector to be not so accurately updated in the vicinity of the optimal solution, which reduces the accuracy of the estimate. Because of that, we made λ change adaptively during the minimization process. Initially, λ is assigned a rather high value in order to speed up the convergence. During the minimization process, λ is progressively decreased in order to adapt the magnitude of the update step to the size of the search space. This step adaptation leads to improved results with a modestly higher computational effort.

Localization—The first rough localization of discontinuities is done by searching for the gradient's maxima over the surface provided by the interpolation step, similarly to Canny's edge detection. This is done by searching for those points whose gradient has an amplitude that exceeds an assigned threshold and is maximum along the gradient's orientation; such points are marked as *edge points*. The threshold is computed in an automatic and adaptive fashion, while taking the status of the interpolation process into account. More precisely, at the k-th iteration, the gradient $G_{i,j}$ in the point (i,j) is compared with a threshold $S_k(i,j)$ that is proportional to its average value in a square neighborhood $I_{i,j}^{(k)}$ of the point (i,j)

$$S_k(i,j) = a \cdot E_{I_{i,j}^{(k)}}[|G(x,y)|]$$

where $E_I[\cdot]$ denotes the mean operator over the set I. As the number of iterations increases, the size of this neighborhood decreases. This way, only the most significant discontinuities (with biggest size and depth difference) are detected at the beginning, while the minor ones are detected in the following steps of the interpolation process. Moreover, the coefficient a increases with the number of discontinuities that have already been detected. This progressive increase of the threshold compensates the increase of the gradient variance due to the decrease in size of the neighborhood $I_{i,j}^{(k)}$, and has shown to effectively improve the ability to reject peaks of gradients which do not correspond to discontinuities. This adaptive thresholding is necessary as, during the initial steps, the oscillations of the surface due to the relaxation process are still of significant magnitude and could otherwise be detected as discontinuities.

The location of the discontinuity curves obtained so far still needs refinement, as the obtained contours are still jagged or interrupted here and there. This problem typically occurs with pixel-based edge-detection methods, where the correlation between neighboring edge pixels is not taken into account. For this reason, the above edge detector is followed by a process of edge analysis and classification, which scans the edge contour and defines a chain of segments that connect only a subset

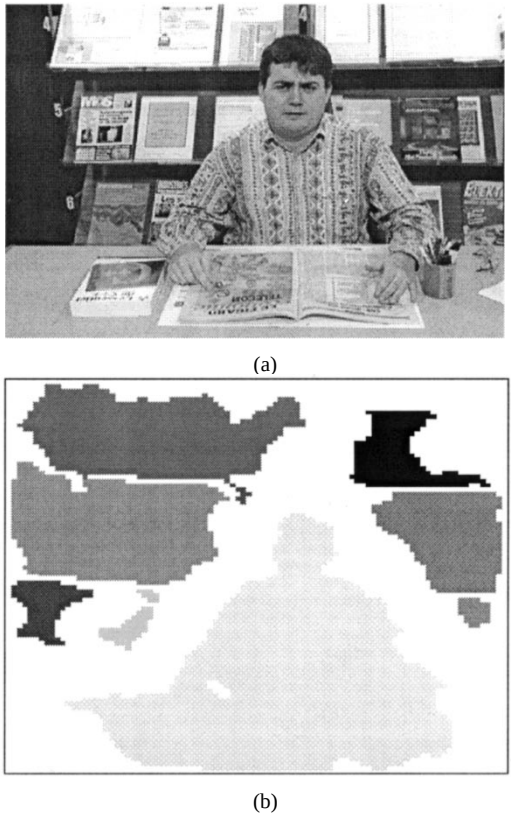
of the edge points, chosen in such a way to maximize the contour's smoothness. The selection of points is done recursively. From the last selected edge point, the next one is chosen as the nearest edge point that lies approximately on the same direction as the last detected points [18]. This operation is performed very efficiently by using a *search mask* which decides the order of preference among neighboring points. This mask is positioned on the current edge point and oriented in the "dominant" direction of the last detected points. The edge point with the highest weight within the search mask is taken as the next point of the processed chain of segments. If no edge points are found within the search mask, the search is stopped, so that it can start with another segment. This approach allows us to freely adjust the edge post-processing algorithm by simply changing the weights of the search mask, so that the method will be able to correct irregularities and make bridges over interruptions in a wide variety of conditions.

Even after the above contour refinement process, the obtained discontinuity curves still do not satisfy some properties that characterize the object boundaries. In fact, they do not lie exactly on the visible boundary of the foreground object; and they are usually not close.[3] The reason why only roughly do they approximate the shape of the foreground objects is that they lie in regions where 3-D data is usually not available. In fact, the discontinuity contours are located at the outermost boundaries of occluding objects, where stereometric principles often fail (lack of stereo correspondences). In principle, using only the available depth information, there is not much else we can do in order to improve the discontinuity location, unless we use additional sources of information, such as the original reference image, as described in the next section.

Segmentation into Objects—As already stated above, the edge detection process does not guarantee that the discontinuity contours will be closed. In particular, in case of wide occlusion regions and small depth differences between facing surfaces, the gradient of the interpolated surface from one object to the other will exhibit a rather modest peak. This makes the object separation a hard task. Typical consequence of this problem is the generation of interrupted discontinuity curves, which leave the surfaces still connected. In order to recover the correct topology of the discontinuity map and, therefore, to separate *occluding* from *occluded* objects, it is necessary to partition the interpolated surface into sub-surfaces that represent different objects. This surface segmentation process should determine "well-connected" surface regions, and should eliminate those thin strips of surface that still connect different regions because of boundary interruptions.

As already stated above, the regions near object boundaries are usually characterized by an absence of 3-D data, as surface self-occlusions prevent boundary features from being stereo-corresponding. The thickness of such "*blind*" regions depends on how far apart the cameras are, with respect to their distance from the scene. In order to determine close boundaries and complete a preliminary segmentation, we can thus "thicken" the boundaries within the blind regions until we are left with well-connected sub-surfaces.

[3]This is true under the assumption that the border of the reference view be considered a closing line for the discontinuities. In other words, discontinuity lines that begin and end at the image border because the enclosed object is only partially visible in the reference view, are considered closed.



(a)

(b)

Fig. 5. An example of pre-segmentation of the interpolated scene. All discontinuities are expanded by the algorithm until a complete separation between surfaces with different depth is complete. Each subsurface, here characterized by a different grey level, represents a different object. (a) Original image, middle viewpoint (reference). (b) Partition into sub-surfaces.

This task is accomplished by means of a region growing algorithm applied to the binarization of the surface map, where the binary information means the *blindness*, that is the absence of surface in that point. More specifically, the algorithm starts from a discontinuity point $p^{(0)}$ and marks all of its neighbors $p_i^{(1)}$ as *candidates* for the expansion of the boundary within the blind region. If the point $p_i^{(1)}$ also belongs to the blind region and has been marked as candidate for the boundary expansion by a sufficient number of discontinuity points, then the boundary is expanded to the point $p_i^{(1)}$. The process is repeated as long as there is boundary growth. In this way, isolated points and thin strips of surface are included in the blind region. At the end, all the connection strips between surfaces that do not contain any input 3-D point, have been separated. An example of the resulting segmentation is shown in Fig. 5.

B. Step 2: Shape Refinement at Object Boundaries

As explained above, the interpolation provides us enough information to determine the topology of the discontinuities, but not their exact location, as the 3-D input points-set is not informative enough. On the other hand, an accurate localization

of the object boundaries in the reconstructed scene is of crucial importance for the final quality of the reconstruction. The object boundaries, in fact, can provide a human observer a great deal of information about the object shape. In order to accurately localize the object boundaries, a most significant additional information source is represented by the luminance profile of the original images, as the *occluding contours* normally originate luminance edges. It is important to point out that the perspective projection onto the reference image and the previously defined perspective depth-map share the same geometry, therefore we should expect a one-to-one correspondence between occluding edge contours (in the reference view) and object boundaries in the reconstructed surface. The luminance edges contained in an image, however, can be classified into three classes:

1) *texture edges*, which are either a characteristic of the surface reflectivity or caused by the illumination conditions (shadows, structured light, specular reflections, etc.);
2) *sharp edges*, which are associated to surface creases;
3) *occlusion edges*, which are the outermost boundaries of the object, where the visual rays are tangent to their surface.

As what we are looking for are edges of the third type, texture and sharp edges can be a source of problems. A significant help, however, is given by our *a priori* knowledge of the approximate position and orientation of the occluding contours, which is provided by the preliminary interpolation/segmentation procedure.

Some methods, in which luminance edges are used for an accurate localization of the object discontinuities, have been proposed in the literature [12], [20]. This problem can be considered, in fact, as a special case of the more general edge localization problem, with some *a-priori* knowledge on the approximate edge location or on the shape of the encircled objects [9], [13], [22]. Similarly to the above-cited methods, our approach to an accurate localization of the object boundaries is here formulated as an optimization problem. In order to do so, we define a *merit function* $\mathcal{F}(\omega)$ of the discontinuity contours, which we want to maximize as the configuration of the discontinuity curves ω varies. The merit function incorporates the distance from a luminance edge and accounts for the smoothness of the discontinuity contour.

As the emphasis of this work is on the computational efficiency, we developed a particularly fast and effective strategy for iteratively updating the shape of the discontinuities ω during the optimization process. In our approach, the points of the discontinuity contours are shifted in such a way to increase the global merit factor and to minimize the likelihood of further shifts of the same point, with the result of minimizing the number of iterations. The details this minimization procedure are discussed in the following.

1) The Merit Function: As anticipated above, the merit function is defined as a linear combination of two terms: one that measures proximity to luminance edges that are likely to represent the occlusion boundaries, and the other one that measures the smoothness of the contour. The use of a linear combination has the two-fold advantage of necessitating a modest computational effort and of deciding which term to favor in quite a straightforward fashion. Given a closed discontinuity curve c, the optimal boundary curve c_{opt} can be determined as

$$c_{\text{opt}} = \arg\min_{c} |\mathcal{F}(c)|$$

where

$$\mathcal{F}(c) = k_G f_{\text{contrast}}(c) + k_S f_{\text{shape}}(c) + k_D f_{\text{update}}(c) \quad (5)$$

where $f_{\text{contrast}}(c)$ is a measure of the image contrast along the considered contour; $f_{\text{shape}}(c)$ measures the shape regularity of the curve; and $f_{\text{update}}(c)$ is a function of the distance between the contour c and its original location before the update. Such terms are weighed by the coefficients k_G, k_S and k_D. A brief discussion on each one of the above terms follows.

Image Contrast—The aim of the operator $f_{\text{contrast}}(c)$ is to have the luminance edge that is most likely to correspond to the object's occlusion boundary "attract" the curve c. Considering the fact that the discontinuity contour c_0, as determined by the interpolation/segmentation procedure, is already a good approximation of the occluding boundary, it is reasonable to expect the true boundary to be located in its proximity and to be approximately oriented in the same direction. The luminance gradient's magnitude is thus likely to exhibit a peak in correspondence to the optimal boundary c_{opt}, where the gradient is oriented about perpendicularly to c. Consequently, in order to shift c toward c_{opt}, the term $f_{\text{contrast}}(c)$ is defined in such a way to comply with these expectations

$$f_{\text{contrast}}(c) = \sum_{(x,y)\in c} |\nabla l(x,y) \sin\{\alpha[c(x,y), \nabla l(x,y)]\}| \quad (6)$$

where $l(x,y)$ is the luminance profile of the reference view and $\alpha[c(x,y), \nabla l(x,y)]$ is the angle formed by the local tangent to c in (x,y) with the direction of the luminance gradient vector, ∇l, in the same point.[4]

When color images are available, the localization of occluding edges can become much more effective. The gradient of the hue color component, for example, is often much more informative than the luminance gradient. Shadows, for example, originate undesired luminance edges, but they cause negligible changes in the hue component.[5]

Several experiments have been carried out over a number of different color scenes, using different color spaces and different definitions of the color gradient in each case. We obtained the most significant results with the RGB and the YUV color spaces. In particular, we achieved our best results in the RGB space by defining the color gradient vector $\nabla C(x,y)$ as the color component of maximum magnitude

$$\begin{aligned} |\nabla C(x,y)| &= \max\{|\nabla R(x,y)|, |\nabla G(x,y)|, |\nabla B(x,y)|\} \\ \angle \nabla C(x,y) &= \angle\{i \in \{R,G,B\} : |\nabla i(x,y)| = |\nabla C(x,y)|\} \end{aligned} \quad (7)$$

[4]Notice that the vertices of the discontinuity curve are defined on a continuous domain, which justifies the use of the notation (x, y).

[5]This is true under the assumption that the scene is illuminated by white light sources only.

$R(x,y)$, $G(x,y)$ and $B(x,y)$ being the three color components. As a consequence, we obtain

$$f_{\text{contrast}}(c) = \sum_{(x,y)\in c} |\nabla C(x,y) \sin\{\alpha[c(x,y), \nabla C(x,y)]\}| \quad (8)$$

Shape Regularity—The goal of the shape regularity term in the merit function is to get rid of local irregularities in the boundary shape, as they are unlikely to correspond to the real object shape. Such irregularities in the boundary contour appear as small "dents" caused by the vicinity of other contours (usually textural edges) that cannot be easily told apart. In order to prevent the formation of such artifacts, we can try to minimize its perimeter. In order to do so, we can define

$$f_{\text{shape}}(c) = -\mathcal{L}(c) \quad (9)$$

where $\mathcal{L}(c)$ is the total length of the object boundary c.

Some other smoothness operators have been defined in the literature, which provide a better measurement of the shape regularity. An example is in [13], which is based on the radius of the local osculating circle. However, in all the experiments that we conducted over a number of test scenes, no significant loss could be detected when using the definition (9) instead of more complex ones. Our choice, in fact, has the advantage of requiring a very limited computational effort.

Update Distance—Let us consider an object whose boundary is not completely visible as, for example, part of it lies in a shaded area. The *image contrast* term is thus unable to shift the boundary toward any edge, while the *shape regularity* term would have the boundary completed with the shortest path that connects the extremes of the interruption. For this reason, a third term was introduced in the merit function, which is meant to keep the updated boundary as close as possible to its original location, by acting like "springs" placed between boundary vertices and corresponding starting points. This update distance factor was expressed in terms of the mean square distance between updated and original vertices, as projected onto the image plane

$$\begin{aligned} f_{\text{update}}(c) &= -\sum_{P_i\in c} d\left(P_i^{(n)}, P_i^0\right) \\ &= -\sum_{P_i\in c} \left[\left(x_i^{(n)} - x_i^{(0)}\right)^2 + \left(y_i^{(n)} - y_i^{(0)}\right)^2\right] \end{aligned} \quad (10)$$

where $P_i = (x_i, y_i)$ are the coordinates of the i-th vertex point of the polyline $\{P_i\}$ that represents the boundary c; P_i^0 is the original location of the vertex; $P_i^{(n)}$ is its location at the n-th iteration of the optimization process; and $d(p_1, p_2)$ is the Euclidean distance between two points. Through the introduction of this last term, the update algorithm will only smoothen the boundary where it is not visible.

2) The Optimization Strategy: In principle, the optimization problem, which consists of maximization of the merit function (5), could be solved through an exhaustive search. However, the number of unknowns makes this approach unfeasible. Moreover, as one the main goals of this work is to maximize the computational efficiency, the definition of a good optimization strategy becomes of utmost importance.

Representation of the Boundaries—The boundaries must be described in such a way that the update process will be able to handle them. Our choice is to use closed *polylines*, which are piecewise linear contours (chains of vertices connected by segments). A boundary is then represented by an ordered list of vertices $\{P_1P_2\cdots P_i\cdots P_L\}$ each of which is described by the triplet (i, j, dir), where (i, j) are the coordinates of the vertex and *dir* is the orientation of the segment that connects the vertex with the following one (this last piece of information is clearly redundant, but it enables a much faster implementation).

The generation of the polyline is done in such a way to minimize the distance from the original boundary. For this reason, our approach is to follow the boundary and place the vertices, whenever possible, in correspondence of corners and points of high curvature.

Boundary Modification—The updating strategy basically consists of a progressive deformation of each boundary through a local shift of each vertex. One key feature of this approach is that a change in the location of one vertex, say P_i, will only affect a limited portion of the boundary, i.e., the two adjacent segments $\overline{P_{i-1}P_i}$ and $\overline{P_iP_{i+1}}$. As a consequence, the merit function will only change in the chain $\overline{P_{i-1}P_{i+1}}$ of such two segments, while it will remain unchanged everywhere else. This allows us to update the merit function, after the motion of P_i, in a differential fashion, which is clearly very efficient from the computational standpoint. The update will be given by the difference between the new and the old merit function over the chain $\overline{P_{i-1}P_{i+1}}$ of two segments. In order to maximize the merit function, the optimization process makes sure that the value of $f_{\text{merit}}(c)$ increases at each iteration, by looking at the sign of Δf_{merit}.

In order to efficiently compute Δf_{merit}, we can derive the updates of each one of the three terms of the merit function

$$\Delta f_{\text{merit}} = \Delta f_{\text{contrast}} + \Delta f_{\text{shape}} + \Delta f_{\text{update}}$$

where

$$\begin{aligned} \triangle f_{\text{contrast}}(c) &= f_{\text{contrast}}\left(P_i^{(n)}\right) - f_{\text{contrast}}\left(P_i^{(n-1)}\right) \\ &= \left|\nabla C\left(P_i^{(n)}\right)\sin\alpha\left(P_i^{(n)}\right)\right| \\ &\quad - \left|\nabla C\left(P_i^{(n-1)}\right)\sin\alpha\left(P_i^{(n-1)}\right)\right| \end{aligned} \quad (11)$$

$C(P_i)$ being the color component of the maximum gradient's magnitude, as defined in (7)

$$\begin{aligned} &\Delta f_{\text{shape}}(c) \\ &= f_{\text{shape}}\left(P_i^{(n)}\right) - f_{\text{shape}}\left(P_i^{(n-1)}\right) \\ &= -\left[\mathcal{L}\left(c^{(n)}\right) - \mathcal{L}\left(c^{(n-1)}\right)\right] \\ &= -\left(\overline{P_i^{(n)}P_{i-1}} + \overline{P_i^{(n)}P_{i+1}} - \overline{P_i^{(n-1)}P_{i-1}} - \overline{P_i^{n-1}P_{i+1}}\right) \end{aligned} \quad (12)$$

and

$$\begin{aligned} \Delta f_{\text{update}}(c) &= f_{\text{update}}\left(P_i^{(n)}\right) - f_{\text{update}}\left(P_i^{(n-1)}\right) \\ &= d\left(P_i^{(n)}, P_i^0\right) - d\left(P_i^{(n-1)}, P_i^0\right). \end{aligned} \quad (13)$$

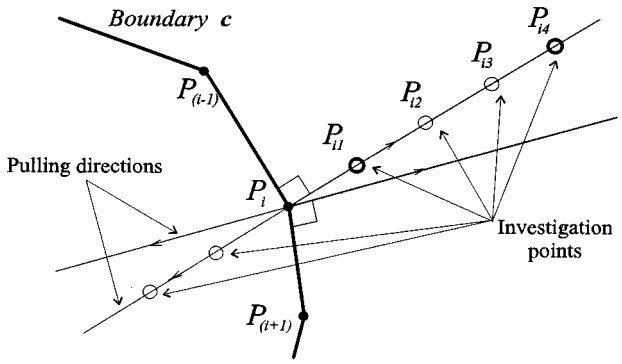


Fig. 6. Possible candidates for updating a boundary vertex.

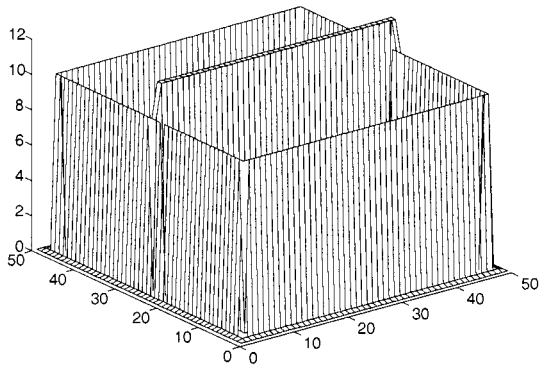


Fig. 7. A synthetic input 3-D point-set.

All such terms can be computed by using no more than three adjacent vertices, which makes the method particularly efficient from the computational standpoint.

As far as our strategy for the contour's shape refinement is concerned, since a vertex shift along the boundary would not significantly change the contour's shape, we chose to move the vertices perpendicularly to the boundary itself. However, as each vertex originates two polyline segments, two different perpendicular directions will be considered. Having two candidate directions instead of one (e.g. the perpendicular to the average local orientation of the boundary) makes the contour more likely to fit the boundary edges, particularly in regions of high curvature, as our tests on real scenes confirm.

As far as the magnitude of the displacement vector is concerned, it is necessary to find a tradeoff between two contrasting needs. On one hand, a small vertex shift leads to a more accurate boundary's repositioning on the object's edge, with a certain risk of being attracted by the wrong edge if the boundary is relatively far apart from the correct one. On the other hand, larger displacements allow the boundary to be repositioned on more distant edges, but the positional accuracy is, in fact, reduced. Furthermore, the update of the vertex location could start oscillating during the iterations. In order to take advantage of both possibilities, we adopted a *multi-step search* approach, which consists of selecting several candidates $(P_{i1}, P_{i2}, \cdots, P_{iN})$ for the shifted location of P_i, scattered along the two specified directions at a progressively larger distance from P_i. For each one of such points, the luminance/color gradient is computed. Among those points that exhibit a gradient of significant magnitude, the one with the best contrast factor is selected, and the vertex is updated with the minimum step along its direction. With reference to Fig. 6, if the best value of f_{contrast} is the one that corresponds to P_{i4}, then the P_i is shifted to P_{i1}. Through this approach, we obtain an accurate vertex positioning even when the edges are quite far away from the original boundary.

Process Randomization—The proposed localization approach, like any iterative optimization procedure with a complex merit function, could converge to a maximum that does not correspond to the global optimum. In order to minimize the risk of encountering a relative maximum, we introduce a partial randomization in the optimization process, in a similar fashion as in methods of *simulated annealing*. This improves the likelihood of a correct convergence, at the cost of a moderately higher number of iterations. In order to do so, the algorithm insists on the same vertices, each time adding a modest random contribution to the merit function. This gives the algorithm a chance of escaping from secondary local maxima, while improving the localization's accuracy.

IV. Experimental Results

In order to test and validate the proposed technique, we conducted two types of experiments. The first series of tests was conducted on synthetically generated *ad hoc* 3-D input data. The point-sets were generated in such a way to make it visible when the reconstructed surface deviates from the *ideal reconstruction*. The aim of such tests is to confirm the absence of systematic reconstruction errors and to evaluate the asymptotic accuracy of the algorithm. A second series of experiments were conducted on real scenes acquired with multiple camera systems and processed by automatic 3-D reconstruction algorithms [29]. The adopted stereometric techniques are based on the automatic detection, matching and back projection of image features, each generating one of the 3-D points of point-cloud. Point-sets generated with matching-based reconstruction algorithms have common properties, for that regards the reliability and *noisiness* of the points, as explained in the following. For this reason, the experiments with real scenes were aimed at confirming the capability of our technique, with such 3-D input data, to successfully segment the 3-D scene into the different scene objects and determine the objects' boundaries with high accuracy. Besides, aim of the experiments was also the evaluation of the computational efficiency of our technique.

A. Reconstruction from Synthetic Data

One of the considered data sets represents a surface characterized by a closed square depth discontinuity contour with a crease in the middle (like a roof). The surface contains both depth discontinuities and gradient discontinuities (creases), which meet in two points. Inside the smooth areas the surface is expected to be planar. The 3-D point-set is made of a number of coplanar points, which are scattered only in the close proximity of the discontinuity contours (see Fig. 7). This particular test surface constitutes a rather difficult test for the detection of discontinuities and provides us with a good benchmark for testing the smoothing capacity of the interpolator within a continuous region.

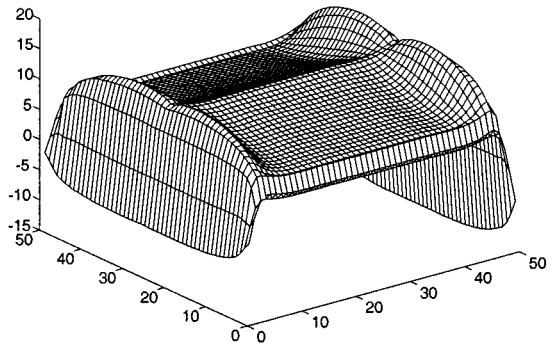


Fig. 8. Pre-interpolation of the data set, assuming that no discontinuities are present.

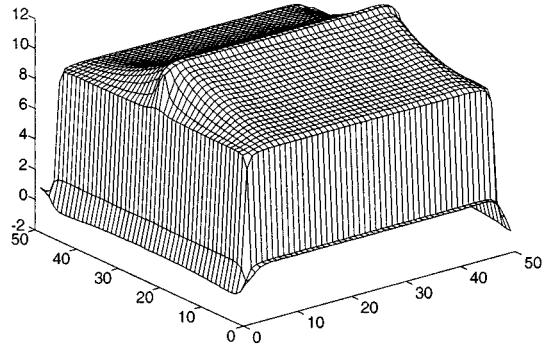


Fig. 9. Surface interpolation under the assumption of continuity but with a modest cohesion force.

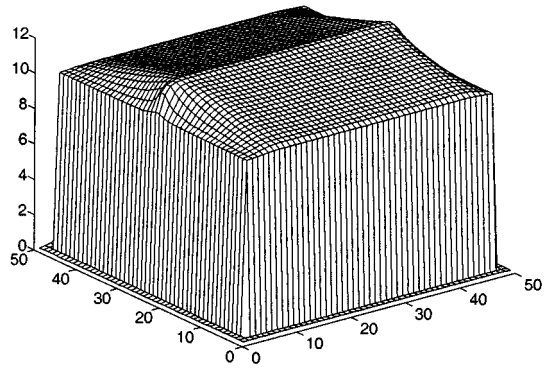


Fig. 10. Surface obtained with the proposed controlled-continuity interpolator.

Fig. 8 shows the result of the pre-interpolation, which generates a single continuous surface, as expected. As we can see, the pre-interpolator behaves exactly as a rigid thin-plate fitting algorithm. Fig. 9 shows the surface obtained using the same procedure, but where only the continuity constraint on the surface gradient has been removed. This way, only the continuity of the surface is preserved. By removing the rigidity constraint, the interpolator emulates a thin plate under tension, which approximates the desired surface much better, thanks to a better behavior at the discontinuities. On the other hand, a non-rigid thin plate could easily lead to results of poor quality, if the input point-set were noisy. This is a situation that typically occurs with 3-D point-sets obtained from real images.

In Fig. 10, the results of the interpolation/segmentation algorithm are shown. As we can see, the obtained surface coincides with the desired one. Furthermore, as shown in Fig. 11, the depth discontinuity contours were exactly localized as well.

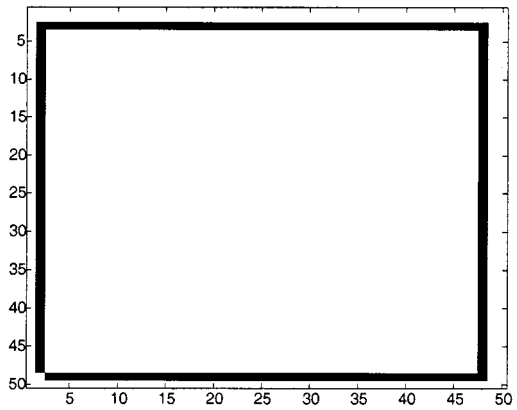


Fig. 11. The obtained depth discontinuity map $\rho(i,j)$.

In order to evaluate the accuracy of the interpolation, the obtained surfaces have been compared with the corresponding ideal reconstructions and the depth differences have been measured throughout the domain. With all the tested data sets, the difference was generally negligible, and, in the neighborhood of the discontinuities, it never exceeded one part over 1000.

B. Reconstruction from Real Data

The synthetically generated 3-D point-sets have been used only to test the first part of our technique, which is the interpolation/segmentation algorithm. When dealing with a multi-camera acquisition system and real scenes, however, the 3-D point-sets obtained through stereo-reconstruction methods present some typical problems:

1) The 3-D points are generally affected by a small, normally negligible, position error, except for a little part of them, called *outliers*, whose position is macroscopically wrong (normally caused by a matching error), and whose distance from the ideal interpolating surface is much larger than the standard deviation of the whole point-set. Because of this significant deviation from the overall statistical behavior, such *outliers* can be easily individuated and eliminated during the interpolation step, by means of statistical analysis of the distance from the currently interpolating surface.
2) There are significantly wide regions in which it is impossible to retrieve reliable 3-D information. This problem is to be attributed to a lack of stereo-correspondences because of surface self-occlusions, and is particularly severe when the distance between cameras is comparable with the distance from the scene. Such "blind" regions are, in fact, positioned at the extreme boundaries of the objects, which makes it impossible to determine the exact location of the discontinuity curves from depth information only. For this reason, the performance of this second part of the technique that is the image-based accurate boundary localization becomes of utmost importance.

In order to show the performance of the proposed technique with such data sets, two examples are presented, in which the data sets present different quality. The first case is quite a simple

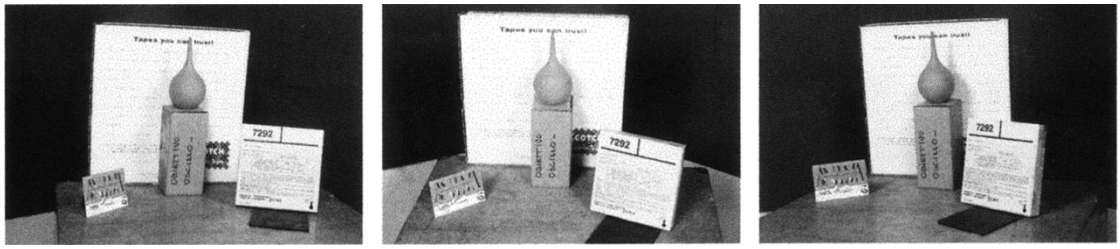


Fig. 12. The three original images of the scene BOXES. The middle view is taken as reference viewpoint.

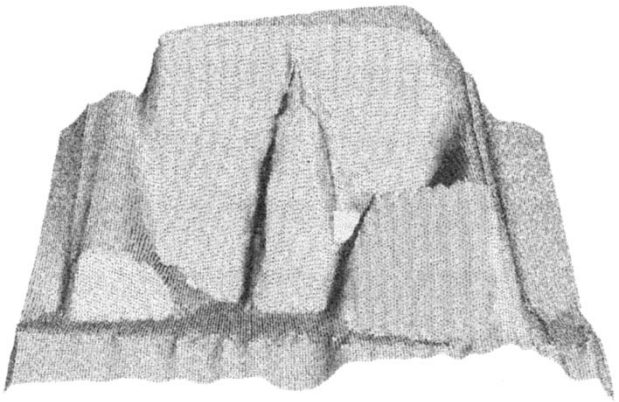


Fig. 13. The 3-D point cloud obtained from the three images, used as input data set.

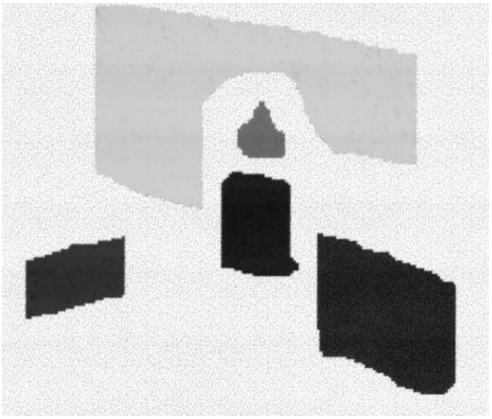


Fig. 14. Map of the segmented sub-surfaces, referred to the reference viewpoint. Each sub-surface actually corresponds to a different object (the surfaces representing the table and the background has been rejected).

scene, in which the reconstruction of the 3-D points has been helped by illuminating the scene with a structured light source. The augmented texture on the object surfaces allows to obtain a denser 3-D point cloud together with a much tinier quantity of outlier points. On the other hand, the second example can be considered a challenge for the proposed method: it represents a typical videoconferencing scene, which has been acquired with a three-cameras system. The surfaces in the scene are more complex in this case, and the available 3-D points are obtained, through area-matching, just by exploiting the intrinsic texture on the object surfaces.

Fig. 12 shows the three original images of the first scene. The middle view is taken as *reference viewpoint*. The area-matching algorithm applied to these images has provided the point cloud shown in Fig. 13, which also is used as initial surface for the interpolation/segmentation algorithm whose results are shown in Figs. 14 and 15. As the figure shows, the segmentation algorithm was able to determine one sub-surface for each object visible in the scene. Starting from these data, the boundary localization step has produced the results shown in Fig. 16, where the detected boundaries has been projected onto the reference image. All the objects have been separated and the boundaries in foreground have been accurately localized, except for the upper part of the pump, which was too thin for the area-matching algorithm to be able to recover its shape.

Fig. 17 shows the original images of the video-conferencing scene, acquired with a trinocular system using a set of three CCIR-601 standard (720 $\times$ 576 pixel) color cameras. The camera system was positioned approximately 2 m from the speaker; the distance between the left and the right cameras was approximately 80 cm; and the middle camera was placed about 30 cm above the other two, therefore this camera was taken as *reference viewpoint*.

As we can see from Fig. 19, the interpolation/segmentation process was able to determine a set of correct and closed discontinuity contours that encircle the various scene objects, using the 3-D point cloud of Fig. 20 as the only input. Nonetheless, the absence of 3-D points in the occlusion regions does not allow the algorithm to locate the object boundaries with sufficient accuracy. The discontinuity contours are, in fact, placed somewhere inside the occlusion regions, as shown in Fig. 20. Such discontinuity contours are the approximate boundaries that are passed to the accurate boundary localization algorithm, together with the original reference view.

The final result of the boundary localization is shown in Fig. 21. Compared with the initial boundaries of Fig. 20, the accuracy improvement is quite visible. In fact, if a segmented sub-surface is smooth and its boundary is visible (typically a scene object), the discontinuity curves will be refined up to pixel accuracy. With reference to Fig. 21, for example, for some scene objects such as the speaker's head, the ball and most of the speaker's body, the distance between the initial boundary and the actual one, is never more than 3 pixels. Furthermore, it is worth emphasizing the importance of *randomization* in the optimization strategy. In fact, a mild randomization of the merit function, combined with a repetition of the iteration step, turned out to introduce significant improvements, particularly where the information provided by the 3-D point-set is poor, and in the proximity of *spurious edges*. Such problems, in fact, tend to generate local sub-maxima of the merit function (see, for example, the upper part of the speaker's head).

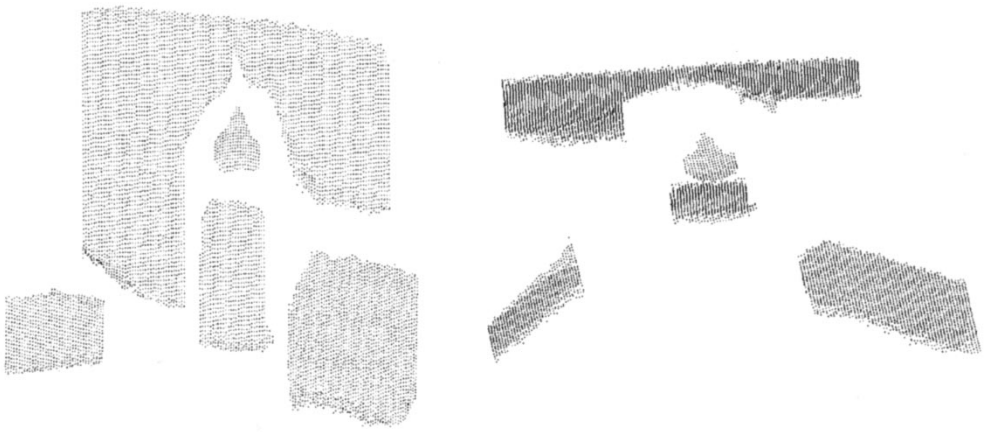


Fig. 15. The 3-D point cloud resulting from the interpolation/segmentation procedure.

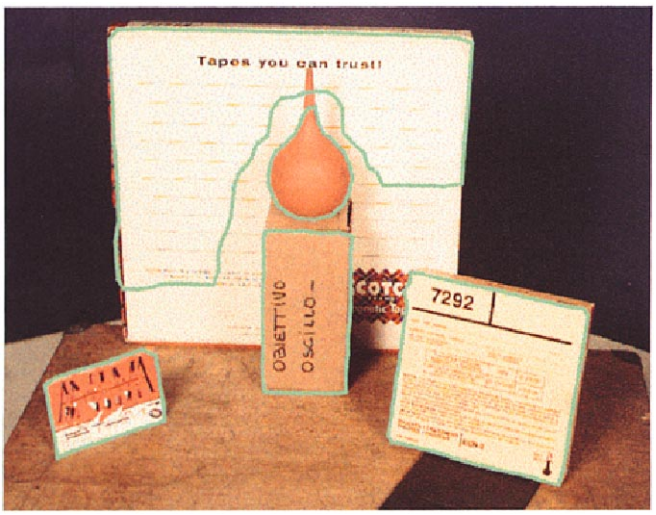


Fig. 16. Scene BOXES: The object boundaries, localized by the boundary refinement procedure, superimposed to the reference image.

Fig. 17. Original images of the scene "Gwen" (left, middle, and right views). The middle view is taken as *reference viewpoint*.

V. Conclusion

In this paper, we proposed a method for accurately reconstructing scene surfaces through a segmentation-based interpolator (*visible-surface reconstruction*), based on the analysis of 3-D point-sets and luminance profiles of perspective scene views.

Key features of this work are the computational efficiency, the robustness of the procedure and the reliability of the results when dealing with real data. Our technique, in fact, is particularly suitable for sparse 3-D data, obtained from automatic stereo-matching methods applied to sets of views acquired by a trinocular camera system.

Our approach consists of two main steps: a pre-interpolation/segmentation phase, and a boundary refinement process. The pre-interpolation/segmentation algorithm segments the observed scene into separate surfaces that pertain different objects, by using the depth map only. Object-boundary refinement is then performed using luminance and/or color information, as provided by the original images. The method we propose turned out to outperform purely variational techniques, especially in the accuracy at the object boundaries. This can be attributed to the adoption of a projective depth-map (instead of an orthographic one) and to the fact that luminance/color edges have been exploited as well.

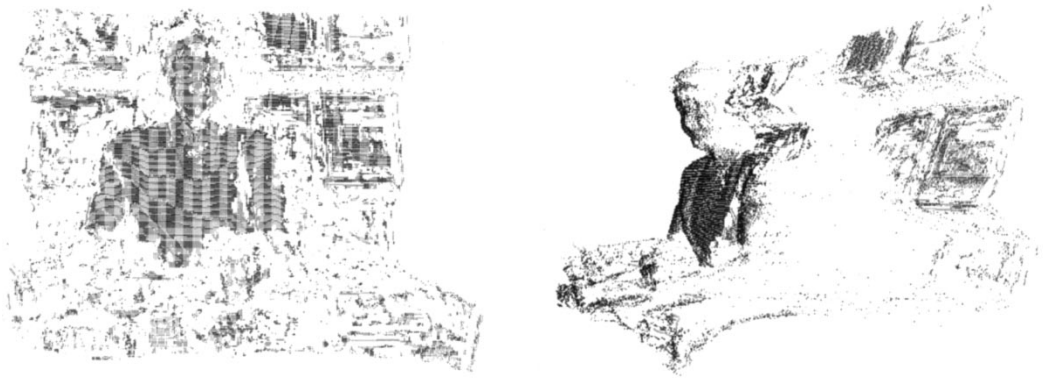


Fig. 18. Two views of the 3-D point-set used by the pre-interpolation/segmentation algorithm. The points were generated by a combined edge-based and area-based 3-D reconstruction technique.

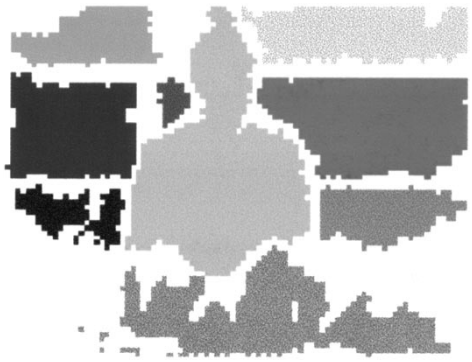


Fig. 19. Map of the segmented object surfaces in the "Gwen" scene.

Fig. 20. Lines of depth discontinuity generated by the interpolation/segmentation algorithm.

Fig. 21. Final result of the boundary refinement.

As far as the computational efficiency is concerned, the required computing effort strongly depends on the size of the interpolation domain, as well as on the number of 3-D points of the data set. For example, the experiment of Figs. 17–21 (TV-resolution images, same resolution for the interpolation domain) was based on a data set of about 9 000 points. In this case, the pre-interpolation/segmentation algorithm took approximately 10 min to be completed on an SGI R-10000 workstation. As far as the boundary refinement is concerned, the computational time varied from less than one minute to some minutes, depending on the situation. Such computation times have to be compared to the time required for a resolution of the functionals with a robust optimization technique, such as a *simulated annealing* approach, which would require a computation time at least one order of magnitude bigger.

In order to further improve the performance of the proposed technique, we are currently working on an extension of our approach to implicit surfaces, which cannot be described by a depth map. We are also working on issues of temporal consistency, in order to exploit the correlation between segmentations at different time instances. The final goal is to implement an automatic processing chain, which able to generate accurate, topologically correct, 3-D, and temporally consistent reconstructions of all the object of the imaged scene.

REFERENCES

[1] J. L. Mallet, "Discrete smooth interpolation," *ACM Trans. Graph.*, vol. 8, no. 2, pp. 121–144, 1989.
[2] D. Terzopoulos, "The computation of visible-surface representation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 10, July 1988.
[3] A. Blake and A. Zisserman, *Visual Reconstruction*. Cambridge, MA: MIT Press, 1987.
[4] W. E. L. Grimson and T. Pavlidis, "Discontinuity detection for visual surface reconstruction," *Comput. Vis., Graph. Image Processing*, vol. 30, no. 3, pp. 316–330, June 1985.
[5] V. Caselles, R. Kimmel, and G. Sapiro, "Minimal surface-based object segmentation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 394–398, Apr. 1997.

[6] J. Duchon, "Interpolation des functions de deux variables suivant le principe de la flaxion des plaques minces," *R.A.I.R.O. Anal. Num.*, vol. 10, pp. 5–12, 1976.
[7] A. N. Tikhonov, "Regularization of incorrectly posed problems," *Sov. Math. Dokl.*, vol. 4, pp. 1624–1627, 1963.
[8] A. N. Tikhonov and V. A. Arsenin, *Solutions of Ill-Posed Problems*. Washington, DC: Winston, 1977.
[9] P. C. Chen and T. Pavlidis, "Image segmentation as an estimation problem," *Comput. Vis., Graph. Image Processing*, vol. 12, pp. 153–172, 1980.
[10] B. Chupeau and E. Francois, "Depth based segmentation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 237–239, Feb. 1997.
[11] L. Najman and M. Schmitt, "Geodesic saliency of watershed contours and hierarchical segmentation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 1163–1173, Dec. 1996.
[12] E. Izquierdo and S. Kruse, "Disparity-controlled segmentation," presented at the Picture Coding Symposium, Berlin, Germany, Sept. 1997.
[13] T. Pavlidis, "Integrating region growing and edge detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 225–233, Mar. 1990.
[14] M. Pardas and P. Salembier, "3-D morphological segmentation and motion estimation for image sequence," *EURASIP Signal Processing*, vol. 38, no. 1, pp. 31–43, 1994.
[15] P. Salembier, "Morphological segmentation for image coding," *EURASIP Signal Processing*, vol. 38, no. 3, pp. 359–386, Aug. 1994.
[16] N. Ahuja, "A transform for multi-scale image segmentation by integrated edge and region detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 1211–1235, Dec. 1996.
[17] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 8, pp. 679–698, June 1986.
[18] J. S.-P. Shu, "One-pixel-wide edge detection," *Pattern Recognit.*, vol. 22, no. 6, pp. 665–673, June 1989.
[19] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 629–639, July 1990.
[20] A. Martelli, "Edge detection using heuristic search methods," *Comput. Graph. Image Processing*, vol. 1, no. 2, pp. 169–182, Aug. 1972.
[21] D. Williams and M. Shah, "Edge contours using multiple scale," *Comput. Vis., Graph. Image Processing*, vol. 51, no. 3, pp. 256–274, Sept. 1990.
[22] Y. Kita, "Elastic-model driven analysis of several views of a deformable cylindrical object," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 1150–1162, Dec. 1996.
[23] A. Jain, Y. Zhong, and S. Lakshmanan, "Object matching using deformable templates," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 267–278, Mar. 1996.
[24] K. M. Lee and C. J. Kuo, "Shape from shading with perspective projection," *Image Understanding*, vol. 59, no. 2, pp. 202–212, Mar. 1994.
[25] R. Franke, "Thin plate splines with tension," *Comput. Aided Geometric Design*, vol. 2, pp. 87–95, 1985.
[26] S. S. Sinha and B. G. Schunck, "A two-stage algorithm for discontinuity preserving surface reconstruction," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, pp. 36–55, Jan. 1992.
[27] D. Geman, S. Geman, C. Griffigne, and P. Dong, "Boundary detection by constrained optimization," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 609–627, July 1990.
[28] R. Manzotti, F. Pedersini, A. Sarti, and S. Tubaro, "Surface Interpolation Techniques for 3-D Reconstruction from Multiple Views,", Int. Rep. of the Image and Sound Processing Group of the Politecnico di Milano, 1996.
[29] F. Pedersini, P. Pigazzini, A. Sarti, and S. Tubaro, "3-D area matching with arbitrary multi-view geometry," *EURASIP Signal Processing: Image Commun.* , vol. 14, no. 1–2, Oct. 1998.

Federico Pedersini was born in Brescia, Italy, in 1965. He received the laurea degree (*summa cum laude*) in electrical engineering in 1991 and the doctoral degree in electrical engineering and communications in 1994, both from the Politecnico di Milano, Milan, Italy.

He spent one year at the Intitut für Theoretische Nachrichtentechnik, University of Hannover, Hannover, Germany, doing research on camera calibration for 3-D reconstruction systems and accurate modeling of camera imaging process. He is currently with the Politecnico di Milano, where he is doing research on image analysis for 3-D reconstruction and measurement, video sequence analysis, and image synthesis.

Augusto Sarti was born in Rovigo, Italy, in 1963. He received the laurea degree (*summa cum laude*) in electrical engineering in 1988 and the doctoral degree in electrical engineering and information sciences in 1993, both from the University of Padova, Padua, Italy.

He worked for one year for the Italian National Research Council doing research on digital radio systems. He then spent two years researching nonlinear system theory at the University of California at Berkeley. He is currently an Assistant Professor at the Politecnico di Milano, Milan, Italy. His research interests are mainly in digital signal processing and, in particular, in video coding, image analysis for 3-D scene reconstruction, audio processing, and synthesis.

Stefano Tubaro was born in Novara, Italy, in 1957. He completed his studies in electrical engineering in 1982.

In 1984, he joined the Politecnico di Milano, Department of Electronics and Information, in 1984, doing research on voice and image coding. In 1986, he joined the Study Center for Space Telecommunications of the National Research Council. Since 1991, he has been an Associate Professor of Electrical Communications at the Politecnico di Milano. His current research interests mainly include digital image processing and, in particular, video-sequence coding at low bit-rate through motion estimation and image segmentation. He also works on stereovision for 3-D scene reconstruction applied to remote manipulation, autonomous vehicle guidance, and telepresence.